PostBL: Post-mesh Boundary Layer Mesh Generation Tool

Rajeev Jain[1] and Timothy J. Tautges[1]

[1]Argonne National Laboratory, Argonne, IL 60439, USA

Abstract. A boundary layer mesh is a mesh with dense element distribution in the normal direction along specific boundaries. PostBL is a utility to generate boundary layers elements on an already existing mesh model. PostBL supports creation of hexahedral, prism, quad and tri boundary layer elements. It is a part of MeshKit, which is an open-source library for mesh generation functionalities. Generally, boundary layer mesh generation is a pre-meshing process; in this effort, we start from a model that has already been meshed. Boundary layers elements can be generated along the entire skin, selected exterior or internal surface boundaries. MeshKit uses graph-based approach for representing meshing operations; PostBL is one such meshing operation. It can be coupled to other meshing operations like Jaal, NetGen, TetGen, CAMAL and custom meshing tools like RGG. Simple examples demonstrating generation of boundary layers on different mesh types and OECD Vattenfall T-Junction benchmark hexahedral-mesh are presented.

1 Introduction

Boundary layers are typically used for CFD applications, in regions with strong gradients – turbulent flow, diffusion type equations, laminar flow, no-slip boundary with strong gradient of velocity and several other cases [1]. In this effort we start from an already meshed model, whereas most other efforts to generate boundary layers insert them before bulk mesh generation. This strategy is particularly useful, when the meshing of the original problem is complicated and involves geometry decomposition and other geometry modifications prior to meshing. A utility for adding layers of elements along the exterior or interior (shared by two materials) boundary is developed. It relies on the edge or surface (2D or 3D) mesh on which boundary layers elements are desired. This method is also useful for creating boundary layer elements in the wake regions along the direction of fluid flow. After boundary layer insertion smoothing can optionally be used to improve mesh quality.

Most boundary layer meshing tools and algorithms treat boundary layer generation as a pre meshing operation. This strategy works well for tri, quadrilateral and tetrahedral meshes, since robust automatic mesh generation for these mesh types is available. For complex hexahedral meshes, which are usually a combination of sweeping and other such techniques, pre-boundary layer methodology is difficult to achieve in cases where decomposition boundaries cross the boundary layers.

Refining techniques stand in the middle of pre and post mesh boundary layer methods, since material and boundary condition definition are applied after completion of refining operation. Also, refining is often not applicable to decomposition surfaces intersected by boundary layers, a combination of surfaces or complex thin boundary models. Tools like CUBIT [2] have problems during refining operation after geometry-based meshing has been done. This is because placement of boundary layer nodes on interior surfaces will disrupt the usual body-associated mesh characteristics of the model. Several issues encountered during pre boundary layer generation are also applicable to post boundary layer tools; Section 2 identifies various contributions and work that has been done in this area.

Section 2 presents background and contributions. Section 3 is MeshKit and libraries used for developing this algorithm. Section 4 summarizes PostBL algorithm. Section 5 explains examples highlighting capabilities and T-junction benchmark problem. Section 6 presents discussion and future work and Section 7 concluding remarks.

2 Background

Many researchers have worked in the field of performance and generation of boundary layer elements for CFD simulations. Application specific as well as generic tools and algorithms have been developed for generating boundary layers. While, significant contributions have been made in the field of tetrahedral mesh generation and hexahedral mesh refinement for boundary layer generation, there has been relatively little work in the area of post mesh boundary layer generation. Botasso et al [3] present a post mesh boundary layer technique for tetrahedral meshes, a mesh motion algorithm is initially used to deflate the original mesh and make room for boundary layer mesh. Further a stack of prisms is inserted and tetrahedronized, their algorithm depends on prism splitting template, calibration and user tuning. PostBL is more general and can be applied to a variety of mesh types.

Most tetrahedral boundary layer mesh generation techniques initially fill up the boundary layer region and then mesh the bulk region. Some methods are based on directional grid refinement procedures for accurate solution of boundary layer and wake flow regions. Bahrainian et al [4] describe an automatic tetrahedral mesh generator that captures boundary layer and wake flow regions and compare the turbulent compressible flow solutions with published data. Pirzadeh et al [5] also describe a new bounding box based meshing technique to generate high quality tetrahedral meshes with boundary layers for aerospace and other applications, they develop VGRIDns, a tetrahedral grid generator developed at NASA Langley Research Center. Garimella et al [6] present a generalized advancing layers method that can be used for non-manifold models. Several basic technical formulations for boundary layer mesh generation, multiple growth curves at boundary layer nodes, node placement, spacing types, checks for element validity at crossover including smoothing, shrinking and deletion operations to avoid element collision are discussed. Focus of this work is on tetrahedral meshes, also decomposed surfaces and hex-based multi-material models are not discussed. Loseille et al [7] study anisotropic mesh adaptation with boundary layer mesh generation. Highlights of generating tetrahedral meshes for automatic adaptive procedures for RANS on complex geometries are presented. Initially, they start from specifying boundary layers as a global mesh generation procedure, later they report problems with the approach and start "local recovery procedures" which is the most robust of the three 'methods' mentioned by the authors. In another paper [8], they discuss various aspects of point insertion methodology including optimization, use of normals and revaluation after each layer insertion. All of the above work can be classified as pre boundary layer mesh generation.

Several publications are reported on generating boundary layers elements in hexahedral meshes. Quadros el al [9] discuss a method for hex mesh generation of thin section solids. Zhang et al [10] describe an octree-based isoconturing method for automatic multi-material tetrahedral and hexahedral mesh generation. Merkley et al [11] present methods and applications of sheet insertion in a hexahedral mesh. Their basic idea relies on dual of quad or hex (2D or 3D respectively). Boundary layer creation by insertion of layers of sheets of hexes along the boundary is described. In their paper about automatic hex mesh generation for turbomachinery applications, Wang et al [12] describe a technique to create boundary layer elements by carefully placing, maintaining and dicing a buffer layer around a geometry. In this work also preserves material and boundary conditions after boundary layer mesh generation process, but it is closely tied to automation and turbo-machinery applications. In another recent study by Maréchal et al [13] improvement to octree-based methods for handling sharp features and generating boundary layers is described.

2.1 Contributions

This paper addresses the problem of generating boundary layers elements when geometry decomposition boundaries that are typically observed during hex meshing cross boundary layer region. Boundary layer generation for multi-material models and wake regions are supported for hex, tet, quad and tri meshes. Multiple surfaces can be specified for boundary layer generation. PostBL caters to the need of generating boundary layer elements for reactor assembly and core meshes. Given an element budget, mesh generation using paving or other such techniques for a large number of rods drilled in a reactor model is difficult, layers of elements can be added to an already setup coarse meshed model. Specifying boundary layers as new materials can generate concentric instrumentation pins or narrow gap regions in reactor cores, which is otherwise hard to generate. The algorithm is implemented using MeshKit design philosophy; interoperability and use of various libraries and packages highlighted here will benefit coupling and extending this work for other related tasks.

3 MeshKit and Implementation

PostBL relies on geometry and mesh libraries developed as part of the Interoperable Tools for Advanced Petascale Simulations (ITAPS) project. The common geometry module (CGM) [14] provides functions for constructing, modifying, and querying geometric models in solid model-based and other formats. While CGM can evaluate geometry from several underlying geometry engines, this work relies mostly on ACIS [15], with an Open Cascade-based [16] version also supported. Finite-element mesh and mesh-related data are stored in the Mesh-Oriented datABase (MOAB) [17]. MOAB provides query, construction, and modification of finite-element meshes, plus polygons and polyhedra. Mesh generation is performed using a combination of tools. CGM and MOAB are accessed through the ITAPS iGeom and iMesh interfaces [18], respectively.

The iMesh concept of sets and tags is important to the implementation of these tools. A set is an arbitrary collection of entities and other sets; a tag is data annotating entities, sets, or the interface. The combination of sets and tags is a powerful mechanism used to describe boundary conditions, material types, and other types of metadata commonly found with mesh.

MeshKit [19] is an open-source mesh library under development at Argonne National Laboratory. It is targeted towards researchers, tool developers and users looking to generate meshes. MeshKit approaches geometry-based meshing and other mesh operations as a di-graph based process. This graph functionality is imported from Lemon graph library [20]. External packages like CAMAL [21], NetGen [22], TetGen [23], MESQUITE [24], Ipopt [25] etc. are implemented as di-graph algorithms for various mesh related tasks. Graph nodes are mesh operations and graph edges are dependencies. Meshing algorithms can be applied to a geometric entity or a collection of entities; operations can also be based on the entities created as a result of the previous operation. This approach uses 2-phase graph execution, setup phase and execute phase. In setup phase, algorithms express their requirements; automatic sub-algorithmic node creation takes place in phase. In execute phase, algorithms actually get executed. Graph based approach allow explicit representation of meshing dependencies which may allow parallelization of the meshing process. This approach also facilitates updating mesh after small changes. Figure 1 is a di-graph for meshing a cylinder geometry. First the surface is quad-meshed using QuadMesh operation, and then Sweep operation fills the cylinder with hexes. Both QuadMesh and Sweep are user specified mesh operations; MeshKit creates automatic graph nodes IntervalMatch, EdgeMesh and MapMesh during the setup phase of QuadMesh and Sweep algorithm. In Figure 1, S1, S2 and S3 are the surfaces, C1 and C2 are the curves and V1 is the vertex forming the cylinder. The type of geometric entity that each mesh operation works on is highlighted inside their respective parallelogram.

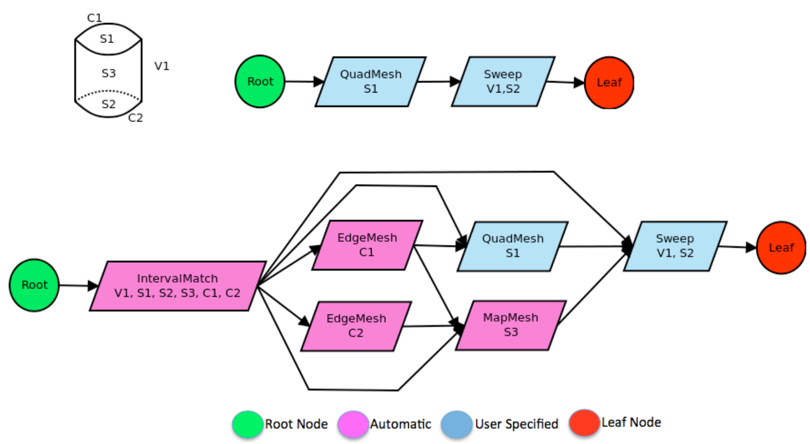


Fig. 1. Di-graph based mesh generation example: sweep-mesh a cylinder.

PostBL mesh operation adheres to this di-graph based approach. PostBL can directly read a mesh file, acting as a stand-alone operation or get the mesh from the previous graph node. Python interface to MeshKit algorithms helps in scripting meshing problems. All meshing algorithms including PostBL can be assessed through MeshKit's python interface based on PyTAPS [26]. Figure 2 shows a user specified di-graph that generates a reactor assembly with boundary layer mesh from scratch. AssyGen [27] operation generates geometry from the text-based input file describing a reactor assembly. This geometry is input to Jaal [28] mesh operation, which feeds into Extrude mesh operation to generate a 3D mesh. Finally, PostBL based on the user specified inputs generates the desired boundary layer elements on the model.

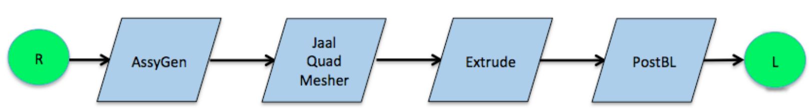


Fig. 2. User specified di-graph for reactor assembly mesh with boundary layers.

4 Algorithm

PostBL is a simple mesh operation implemented as one of the di-graph based algorithms in MeshKit. The core algorithm is as follows:

1. Inputs to the algorithm are mesh and boundary layer variables (see Appendix A for a sample input file). First, mesh dimension is determined and boundary elements and nodes are collected.

2. Check for the existence of material(s) set to which newly created elements must be added. If no material set is specified, add the newly created elements to the material id of its corresponding boundary element.
3. Set the sizes of adjacency and connectivity arrays based on the type of the newly created elements.
4. Loop through all the boundary nodes
    a. Find adjacent entities of same and higher dimension. If it is multiple material case, filter entities that do not take part in boundary layer generation.
    b. Compute normal direction for node creation
    c. For collision detection, check specified thickness with minimum adjacent edge length.
    d. Create boundary layer nodes based on thickness, bias and interval.
    e. Populate connectivity of higher dimensional entities: modify bulk elements and set for new boundary layer elements.
    f. Check for elements that have only an edge or node (2D or 3D) on the boundary, populate their connectivity with newly created node.
5. Loop through all the boundary layer elements (edges for 2D, surfaces for 3D).
    a. Set the connectivity of bulk mesh affected by boundary layer elements computed in 5e.
    b. Create boundary layer elements based on connectivity computed in 5e.
    c. Associate new elements with geometry (if available).
    d. Check for element quality and collision with bulk mesh.
6. Set ‘fixed’ tag for optional mesh smoothing operation after PostBL operation.
7. Report clock, CPU time and save the final mesh
8. Once the final mesh is saved, smoothing can optionally be applied using MESQUITE mesh operation in MeshKit.

Several special cases are encountered during normal computation in step 4b, which generally is the average normal of all the adjacent boundary layer entities. At model boundaries for single material case normal direction is along the boundary that is farthest from the boundary layer node. This choice helps avoid material/volume modification during the boundary layer generation process. When a boundary layer node is part of multiple materials, normals are created along each material boundary to maintain constant material volume.

All mesh file format supported by MOAB [17] can be used as input/output file. Boundary layer can be specified as a surface/edge id (3D or 2D) or as a collection in the form of a NEUMANN set. Several other inputs are problem specific and will be discussed in the results section. All these keywords can be specified using an input file or as a variable in python script. PostBL mesh operation can be run as a C++ MeshKit executable or as a python task using MeshKit’s python interface.

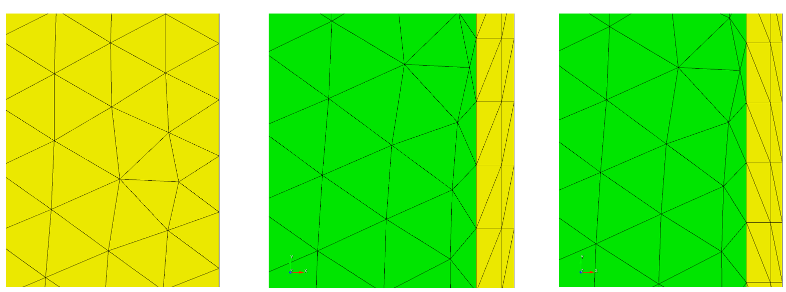
After PostBL, smoothing can be optionally be applied to improve mesh quality. Smoothing can be local or global based on which entities get specified for smooth. Local smoothing is when fixed tag is applied on all nodes except nodes near boundary layer region. Local refining helps in obtaining gradual change in mesh size from boundary layers to bulk mesh. Choice of fixed nodes must be carefully done to preserve the boundary layer region, smoothing the boundary layer nodes may result in loss of bias and boundary elements created by PostBL. Global smoothing operation is when nodes in material boundaries are fixed and the smoothing algorithm can change all interior nodes.

5 Results

Simple and complex examples are carefully chosen to demonstrate the features and limitation of this tool.

5.1 Simple Tri Mesh

Figure 3 shows a very simple tri mesh with boundary layers added on the right in Figure 3(b) and (c). The orientation of the tri elements on the boundary layer elements can be slanting upwards or downwards based on requirement of the simulation. TriScheme variable of 0 or 1 can be specified for generating these results. In figure 3, the triangles in the penultimate layer of boundary layer triangles are shrunk and form a set of large aspect ratio elements. At this point there is no element deletion capability built into this tool. This assumption is true for most hex meshed CFD models, since only a fraction of edge length is typically used for boundary layer generation. When boundary layers collide with bulk mesh, elements are still created and can be restored during smoothing phase. During the generation of boundary layers, jacobian of modified bulk elements is checked for collision and validity of the mesh. If jacobian is less than the minimum specified tolerance, boundary layer generation process stops.

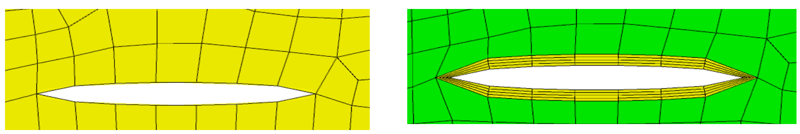


(a) (b) (c)

Fig. 3. Tri mesh example: (a) Original mesh. (b) Two boundary layer on right (slanting downwards). (c) Two boundary layer (slanting upwards).

5.2 Exterior Boundary Layers on a Quad Mesh

Figure 4 shows a simple plate with elliptical cavity and its corresponding boundary layer elements. At the two corners of the cavity the quality of the elements is poor. Several boundary layer algorithms create multiple normal at points where normal from neighboring elements differs by a certain percentage. Since, this is a post mesh operation, the original mesh is responsible for capturing the boundary features of the model, in this algorithm average normal of all the boundary layer elements is chosen for creation of new boundary layer elements. Choice of creating only one normal from each boundary layer node is accurate for most reasonable meshes that have the boundary layer surfaces resolved. It must be noted that the use of tuck or wedge type node placement would better account for local curvature observed here.

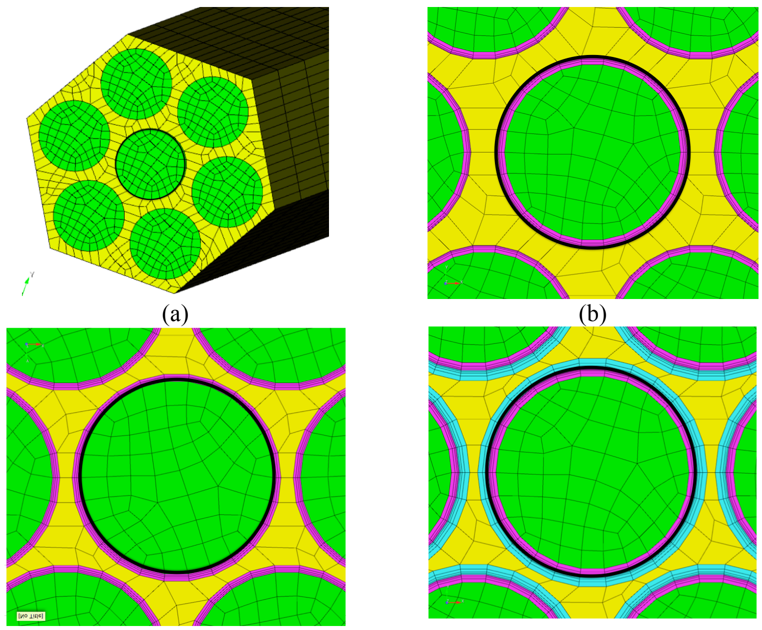


(a) (b)

Fig. 4. Quad mesh example: (a) Original mesh. (b) Mesh with 4 boundary layers.

5.3 Hex Meshed Reactor Assembly

Figure 5 (a) shows a hex meshed nuclear reactor assembly containing six fuel rods and one control rod at the center. Solid line on the control rod indicates the original geometric body. Layers of boundary elements can be generated and optionally setup as a new material type. Setting boundary layers as new material can be useful for creating instrumentation pins and fluid gaps concentric to the original rods. Several types of reactor assemblies have a lot of fuel rods drilled on it and it is hard pave the surface of the geometry, complexity increases as the number of concentric rods to the fuel rods increase. Also, due to the high temperature of the fuel rod, heat flux near the fuel rods is high, thereby requiring finer mesh in this region. In Figure 5, side surfaces of all cylindrical rods are setup as input boundary layers. Three different types of boundary layer meshes can be generated as shown in Figure 5 (b), (c) and (d). Based on the requirements of Fluid Dynamics, Neutronics or Structural Mechanics varying combinations on interior or exterior boundary layers can be generated. In Figure 5(b) boundary layer elements are generated towards the cylindrical region, whereas in Figure 5(c) boundary layer elements are generated away from the cylindrical region. Bulk mesh that does not change during the boundary layer creation can be specified using the FIXMAT keyword. If not specified boundary layers are generated on both sides. Different thickness and bias can be specified for each side of the boundary, a bias of 0.7 is used for this example. Figure 5(d) shows boundary layer elements generated on both sides of the cylindrical region. This feature is useful for creating boundary layer meshes in wake regions where boundary layer is surrounded on both sides by materials.

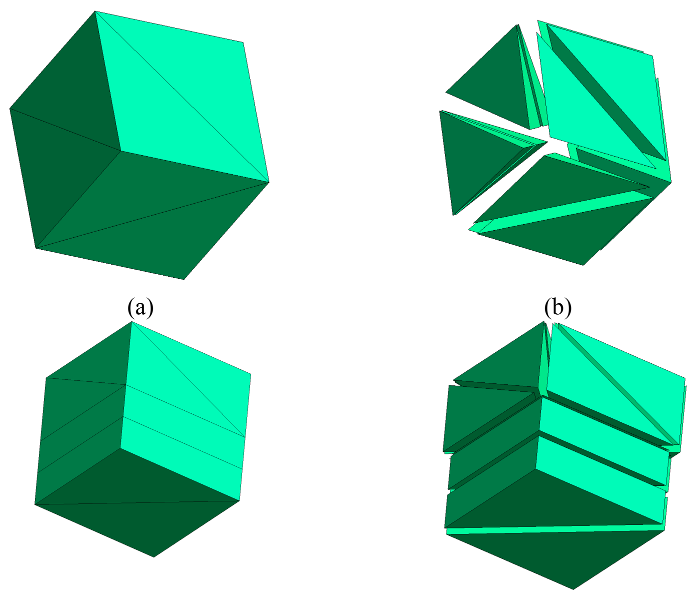


(a) (b) (c) (d)

Fig. 5. Quad mesh example with elements on both sides: (a) Original mesh. (b) Mesh with 4 boundary layers towards the cylinder (c) Mesh with 4 boundary layers away from the cylinders. (d) Boundary layers on both sides of the cylinders.

5.4 Simple Tetrahedral Mesh

Figure 6(a) and (b) show 12 tetrahedral elements on a cube geometry; Figure 6(c) and (d) highlight the hybrid mesh (tetrahedral and prism elements) with 4 prism elements added to the existing mesh. Gmsh [29] is used to display the shrunk elements in this model.



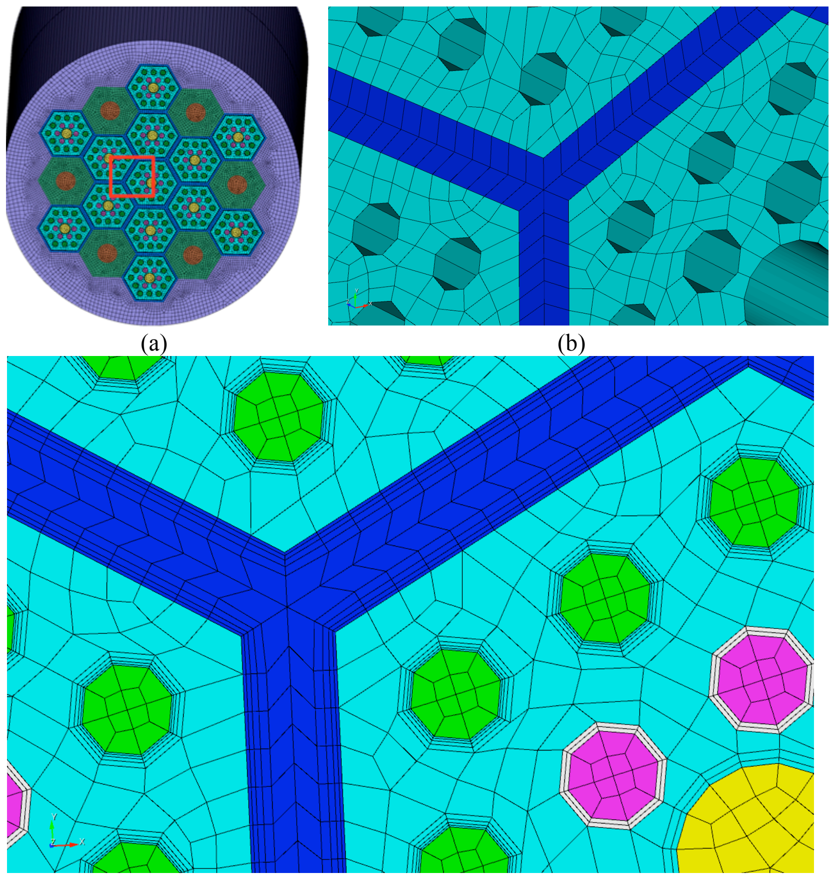
(a) (b)

(c) (d)

Fig. 6. Tet mesh example: (a) Original mesh. (b) Original mesh with elements shrunk by a factor of 80%. (c) Mesh with two boundary layer prisms. (d) Final mesh shrunk by 80% showing tetrahedral and prism elements.

5.5 19 Assembly Hex Meshed Reactor Core

Figure 7(a) shows a reactor core created using RGG tools: AssyGen and CoreGen [27]. Based on the user specified description, AssyGen first creates the two types of reactor geometries involved in the model. Outer covering along with the two assemblies are a meshed separately using MeshKit algorithm(s). Finally, CoreGen is used to copy/move/merge the three mesh files and form the core mesh. Four layer of boundary layer elements are then added to the fluid region and gap region between the assemblies. Actual reactor core models have hundreds of assemblies forming the reactor; adding boundary layers during the generation of core mesh is difficult due to the complexity and the number of parts involved in the model. PostBL preserves the initial material and boundary conditions prescribed by RGG and also allows for creation of varying boundary conditions for different meshes used in a multi-physics simulation. Figure 7(b) shows the close-up view of the region highlighted in Figure 7 (a), note that only gap and fluid region are shown in Figure 7(b). Figure 7 (c) shows the close-up view of the mesh after PostBL operation, in addition to fuel and gap regions, fuel pins are also shown to clearly identify the change from the initial mesh.



(a) (b) (c)

Fig. 7. 19 assembly reactor core mesh: (a) Original mesh. (b) Close-up of original mesh showing fluid and gap regions. (c) Close-up of original mesh showing boundary layers on fluid and gap regions.

5.6 Hex Meshed T-Junction Benchmark Problem

Figure 8(a) shows the OECD Vattenfall T-Junction benchmark mesh that was used in a blind benchmark of various CFD codes. The Argonne code Nek5000 participated in this benchmark using this mesh [30]. The mesh was generated with CUBIT 10.2, then modified to extend the (top) inflow and (side) outflow pipes. Non-circular surfaces are Neumann boundary layers, three layers of boundary elements are added. Figure 8(b) shows zoomed-in view of the T-junction in the original mesh. PostBL uses only one normal direction on this T-intersection point to create new boundary layer elements; this allows for creation of a smooth mesh with similar sized elements. Shape quality metric of this mesh is shown in Figure 8(c), lower but acceptable quality elements are on the boundary due to low aspect ratio of the mesh. Close-up shown in Figure 8(d) shows three new layers as a separate material.

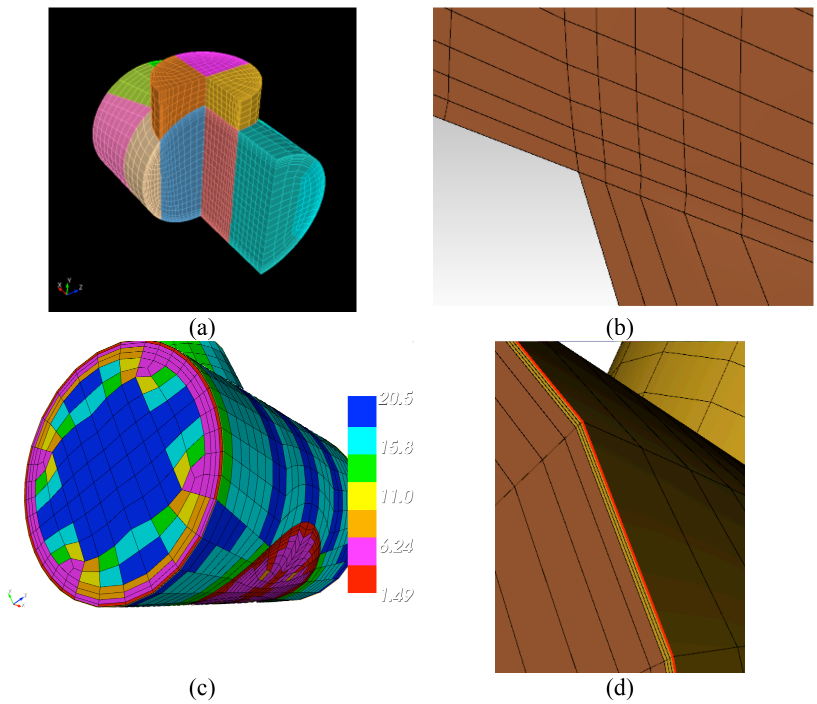


Fig. 8. Hex meshed T-junction example: (a) Original mesh. (b) Close-up of original mesh intersection of cylinders. (c) Mesh with 3 boundary layers showing shape quality metric, smoothened using MESQUITE. (d) 3 boundary layers mesh close-up.

6 Discussion and Future Work

PostBL is useful for CFD analysts studying solution parameters by varying the number of boundary layers; re-meshing of the entire model would be required if pre boundary layer methods are utilized. This could be time consuming and unstable for complicate problems. Unlike some boundary layer tools PostBL does not resolve the boundary layer regions, it adds more layers and provides better resolution, it is desired that the mesh input to PostBL resolve the geometry around the boundary. When there is no collision with other boundaries, high quality boundary layers can be generated using PostBL. Final mesh produced by PostBL can be smoothened to improve the mesh quality at the interface between the bulk and new boundary layer mesh. In future generic templates for refinement of hex, tri and quad meshes during the post mesh boundary layer operation can be added to provide better geometry resolution capabilities. Further research on utilizing the quality metrics of modified bulk mesh for point insertion and normal calculation can be done.

7 Conclusion

PostBL is very well suited for complicated geometries that are difficult to mesh using automatic mesh generation techniques. It presents an attractive alternative to conventional pre boundary layer tools. PostBL can generate boundary layers on interior (shared by two materials) or exterior surfaces of an already existing mesh model. Material and boundary information of the initial model are preserved and extended to include boundary layer elements in the final model. It is ideally suited for hexahedral meshes where decomposition boundaries intersect boundary layer regions, several refinement tools like CUBIT solely rely on volume/surface of the model for boundary layer mesh generation, and this approach does not work with decomposed geometry. Reactor assembly and core meshes have been applied to use PostBL; it can aid increasing the mesh resolution in coolant flow and wake regions. PostBL can be applied to hex, tet, tri, and quad meshes.

Acknowledgement

We thank the Fathom group at Argonne, who maintain the libraries required by this tool. This work was supported in part by the US Department of Energy Office of Nuclear Energy Nuclear Energy Advanced Modeling & Simulation (NEAMS) Program; by the US Department of Energy Office of Scientific Computing Research, Office of Science; and by the US Department of Energy's Scientific Discovery through Advanced Computing program, under Contract DE-AC02-06CH11357.

References

1. Schlichting, Herrmann, Gersten, Klaus Translated by Mayes, C, *"Boundary Layer Theory"* McGraw-Hill, New York, 1987.

2. Sjaardema GD, Tautges TJ, Wilson TJ, Owen SJ, Blacker TD, Bohnhoff WJ, Edwards TL, Hipp JR, Lober RR, Mitchell SA (1994) CUBIT mesh generation environment, users manual, vol 1. Sandia National Laboratories, Albuquerque
3. Bottasso, C. L., & Detomi, D. (2002). A procedure for tetrahedral boundary layer mesh generation. *Engineering with Computers*, *18*(1), 66-79.
4. Bahrainian, S. S., & Mehrdoost, Z. (2012). An automatic unstructured grid generation method for viscous flow simulations. *Mathematics and Computers in Simulation*.
5. Pirzadeh, S. Z. (2010). Advanced Unstructured Grid Generation for Complex Aerodynamic Applications. *AIAA journal*, *48*(5), 904-915.
6. R. V. Garimella and M. S. Shephard, "Boundary layer mesh generation for viscous flow simulations," *International Journal for Numerical Methods in Engineering*, vol. 49, no. 1–2, pp. 193–218, 2000.
7. Loseille, A., & Löhner, R. (2013). Robust Boundary Layer Mesh Generation. In *Proceedings of the 21st International Meshing Roundtable* (pp. 493-511). Springer Berlin Heidelberg.
8. Loseille, A., & Löhner, R. (2009). On 3D anisotropic local remeshing for surface, volume and boundary layers. In *Proceedings of the 18th International Meshing Roundtable* (pp. 611-630). Springer Berlin Heidelberg.
9. Quadros, W. R., & Shimada, K. (2002). Hex-layer: layered all-hex mesh generation on thin section solids via chordal surface transformation. In *Proceedings of 11th international meshing roundtable* (pp. 169-180)
10. Y. Zhang, T. J. R. Hughes, and C. L. Bajaj, "An Automatic 3D Mesh Generation Method for Domains with Multiple Materials," *Comput Methods Appl Mech Eng*, vol. 199, no. 5–8, pp. 405–415, Jan. 2010.
11. K. Merkley, C. Ernst, J. F. Shepherd, and M. J. Borden, "Methods and Applications of Generalized Sheet Insertion for Hexahedral Meshing," in in *Proceedings of the 16th International Meshing Roundtable*, M. L. Brewer and D. Marcum, Eds. Springer Berlin Heidelberg, 2008, pp. 233–250.
12. Wang, F., & di Mare, L. (2013). Automated Hex Meshing for Turbomachinery Secondary Air System. In *Proceedings of the 21st International Meshing Roundtable* (pp. 549-566). Springer Berlin Heidelberg.
13. Maréchal, L. (2009). Advances in octree-based all-hexahedral mesh generation: handling sharp features. In *Proceedings of the 18th International Meshing Roundtable* (pp. 65-84). Springer Berlin Heidelberg.Xie, Z. Q., Sevilla, R., Hassan, O., & Morgan, K. (2013). The generation of arbitrary order curved meshes for 3D finite element analysis. *Computational Mechanics*, *51*(3), 361-374.
14. Tautges TJ (2005) CGM: a geometry interface for mesh generation, analysis and other applications. Eng Comput 17:486–490
15. Spatial Website (2010) http://www.spatial.com/
16. Open CASCADE Technology Website (2000–2010) http://www.opencascade.org
17. Tautges TJ, Meyers R, Merkley K, Stimpson C, Ernst C (2004) MOAB: a mesh-oriented database, SAND2004-1592. Sandia National Laboratories, Albuquerque
18. Ollivier-Gooch C, Diachin LF, Shephard MS, Tautges T (2007) A language-independent API for unstructured mesh access and manipulation. In: Proceedings of the 21st international symposium on high performance computing systems and applications, IEEE, p 22
19. MeshKit: http://trac.mcs.anl.gov/projects/fathom/browser/MeshKit
20. LEMON (C++ library), *Wikipedia, the free encyclopedia*.

21. CAMAL - The CUBIT Adaptive Meshing Algorithm Library, Sandia National Laboratories.
22. NetGen – automatic mesh generator, Johannes Kepler University Linz, 2008.
23. H. Si. On refinement of constrained delaunay tetrahedralizations. Proceedings of the 15th International Meshing Roundtable, 2006.
24. Knupp, P. (2006, September). Mesh quality improvement for SciDAC applications. In *Journal of Physics: Conference Series* (Vol. 46, No. 1, p. 458). IOP Publishing.
25. Wächter, A., & Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, *106*(1), 25-57.
26. PyTAPS, website, https://pypi.python.org/pypi/PyTAPS/
27. T. J. Tautges and R. Jain, "Creating geometry and mesh models for nuclear reactor core geometries using a lattice hierarchy-based approach," *Engineering with Computers*, vol. 28, no. 4, pp. 319–329, Oct. 2012.
28. C. S. Verma and T. Tautges, "Jaal: Engineering a High Quality All-Quadrilateral Mesh Generator," in *Proceedings of the 20th International Meshing Roundtable*, Springer Berlin Heidelberg, 2012, pp. 511–530.
29. Geuzaine, C., & Remacle, J. Gmsh: a three-dimensional finite element mesh generator with built-in pre-and post-processing facilities. Liege, Belgium, 2009.
30. Obabko, A. V., Fischer, P. F., Tautges, T. J., Karabasov, S., Goloviznin, V. M., Zaytsev, M. A., ... & Aksenova, A. E. (2011). *CFD validation in OECD/NEA t-junction benchmark* (No. ANL/NE-11/25). Argonne National Laboratory (ANL)

APPENDIX

A. Sample .inp file specifying the parameters required by PostBL tool:

```
! Name of the input mesh file
MeshFile 5b.cub

! id of the neumann set on which boundary layer needs to be created.
NeumannSet 55
!Comment line: To specify surface use either Surfaces or NeumannSet keyword.

! id of the surface on which boundary layer needs to be created (commented).
!Surfaces 11

! Material id for newly created hexes will be assigned, default: 999
Material 55

! Boundary layer thickness.
Thickness 0.3

! Number of layers.
Intervals 2

! A bias b/w different layer of boundary layer is always greater than zero.
Bias 1.0

! Name of output mesh file, can be any format that is supported by moab.
Outfile 5bout.h5m

! This marks the end of input file for boundary layer generation.
debug 1
END
```